SN Computer Science (2020) 1:348
https://doi.org/10.1007/s42979-020-00360-3

ORIGINAL RESEARCH

Implementation of Newton's Algorithm Using FORTRAN

Shahida Anusha Siddiqui[1] · **Ali Ahmad**[2]

Received: 12 August 2020 / Accepted: 1 October 2020 / Published online: 17 October 2020

Abstract
A lot of software today dealing with various domains of engineering and life sciences have to deal with non-linear problems. In order to reduce the problem to a linear problem, a lot of state of the art solutions already exist. This work focus on the implementation of Newton's Algorithm (also known as Newton's method), to determine the roots of a given function within a specific user defined interval. The software for this implementation is FORTRAN. Even though FORTRAN is considered to be outdated, it still has a lot of application due to its long history and the existing legacy code. The code is written in such a manner that a user can provide a function and a specific interval and the code should in turn run iterations over the interval and should display all the possible roots within that interval. The results are compared at the end for their accuracy. The program is successful in finding out all the roots within an interval.

Keywords Newton's Algorithm · FORTRAN · Roots of a function · Non-linear problems

Introduction

FORTRAN programming language has been one of the earliest of its kind to be in use for the purpose of writing programs. In that regard, learning it is necessary to have a better grasp on it, in order to have a better understanding of those programming languages that followed. As a learning exercise, it was asked to make a program on FORTRAN, that uses Newton's method as its base to approximate the roots of a function over a fixed interval (both the function and the interval are to be given by the user). The sequence is as follows:

- The theory behind Newton's method will be discussed.
- The algorithm for Newton's implementation will be described.
- Comparison of the program's results with actual data.

We examine three variations of the strategy initiating from Newton's strategy for discovering the roots of a function of a sole variable: the method in higher dimensions, higher order method, and continuous method. A brief account of the advancement of Newton's strategy is given and the inventiveness of this article is not claimed; accentuation is put on applications of Newton's strategy in abstract analysis, in specific, subjectivity of functions between finite dimensional Banach spaces [1].

Imperative hypothetical outcomes on Newton's method regarding the convergence properties, the error estimates, the numerical stability and the computational complexity of the algorithm were assessed. In Newton or Newton Raphson strategy an arrangement of the nonlinear condition $F=0$; where $F0$ is the Frechet derivative of F and X and Y are Banach spaces. In case F may be a real function the geometric interpretation of the Newton method is well known [2].

The proposed strategy is demonstrated to be productive with the assistance of the numerical performance and comparison. The first one, for finding roots of scalar functions, is the numerical comparison between the new Newton formulas, Newton's method and a third order Newton method.

This article is part of the topical collection "Computational Statistics" guest edited by Anish Gupta, Mike Hinchey, Vincenzo Puri, Zeev Zalevsky and Wan Abdul Rahim.

✉ Shahida Anusha Siddiqui
S.Siddiqui@dil-ev.de

Ali Ahmad
aliahmad9292@gmail.com

[1] Technical University of Munich (TUM), Germany; DIL-Deutsches Institut für Lebensmitteltechnik e.V. (German Institute of Food Technologies), Quakenbrück, Lower Saxony, Germany

[2] Suckfüll Holzbau Systeme GmbH & Co. KG, Nieheim, Germany

We observe that the offered algorithm is effective for one dimensional real function [3].

Theoretical Background

Like many other root-finding methods, Newton's method, also known as Newton Raphson method, is a mathematical technique to find the best possible vales (roots) of a real-valued function. For many simpler equations (e.g. linear, quadratic), there already exists set of formulas to calculate the exact roots of an equation. But in cases where the equations are far more complex, this method is very useful to get quick and accurate results.

From the development of Newton's strategy to the higher dimensional analogue (our first variation of Newton's method) and to the newton's applications isn't as blunt. Why ought to one consider as it were the constant and the linear term as an approximation of F, in the Taylor expansion of F? Be that as it may, we can utilize Newton's method to show that the condition ($F=0=y$) declares a solution [1].

The first and second derivative of F are signified by $F0$ and $F00$ in the sense of Frechet. An over the top amount of extensions and variants of outcomes arose in the literature. The Kantorovich theorem, by its sheer significance and by the original and powerful proof technique could be a show-stopper. Intensive research on Newton and related strategies were started by the results of Kantorovich and his school [2].

The Newton–Kantorovich theorem also alluded to as convergence theorem, was set up in 1948, Kantorovich. An inaccurate newton strategy was suggested by Dembo et al. [4]. This technique roughly answers the linear equation. Then convergence theorem of Newton's strategy for distinctive cases was set up by Fourier, Cauchy, and Fine. Newton initially utilized the Newton iteration to illuminate a cubic equation in 1916 [3].

Figure 1 gives a basic explanation of how the Newton's method works.

- at the start the root is approximated
- the value of the function is calculated at that approximation
- the difference (delta y) between x-axis and the point at the graph is calculated.
- if delta y is greater than 0, then the slope of the function at the first approximations is calculated
- the point where the extended line of the slope cuts the x-axis is the next approximation

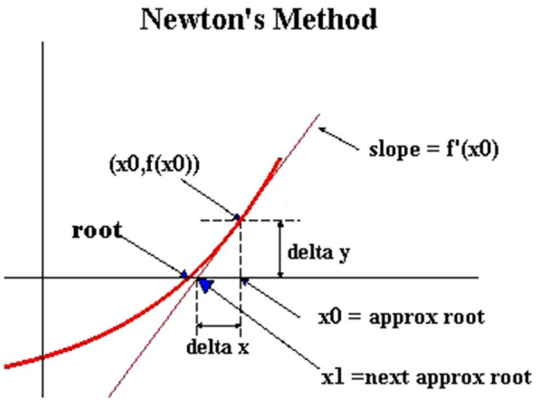

Fig. 1 Graphical representation of Newton's method. Source: Adapted from [5]

- if the difference between the results of the iterations is almost the same, we can assume that we have reached the root of the function.

The above process can also be written in the following mathematical expression:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

And the general form of this process to find the nth approximation is:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}.$$

Constraints of Newton's Method

Newton's method is considered to be one of the most fastest methods to give the results, since it normally requires fewer iterations to reach the results. Even so, there are some conditions that must be mentioned, where this method does not perform so well.

(a) Function with no roots

One of the most common case, where this method will fail is the no-roots case. The iterations keep on

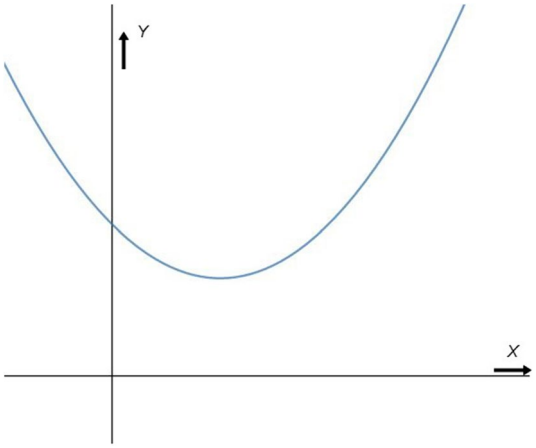

Fig. 2 Case with no possible roots. Source: Adapted from [6]

running for ever since whenever it finds the value of the function and analyses it, it is never close to zero hence the iterations keeps on finding the next possible root. The figure below can display why this can be a problem (Fig. 2).

(b) Problems with the derivative

In order to find the next approximation, the method not only needs the value of the function but also of its derivative. In the formulas above for the calculation of the next approximation, it is shown that the value of the derivative lies in the denominator. So, whenever the value of the derivative is zero (straight line), this means that it can never intersect with the x-axis hence the iteration cannot proceed further. On the other hand, a problem can also arise if the value of the derivative is too large. Because this the next iteration will give the same result as the previous one and hence the process will not move in any direction. The following figure shows the case for zero slope (Fig. 3).

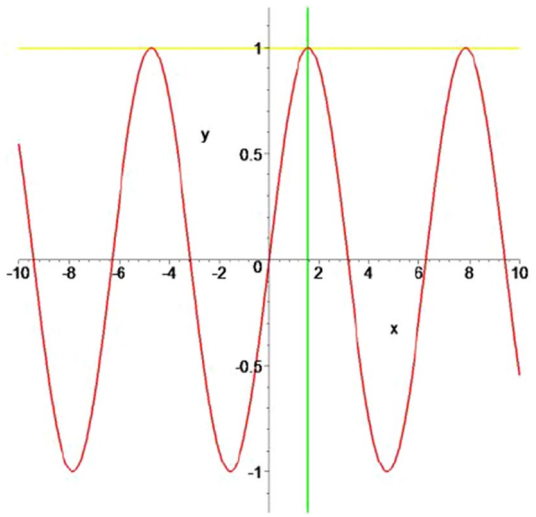

Fig. 3 Function having zero-slope. Source: Adapted from [7]

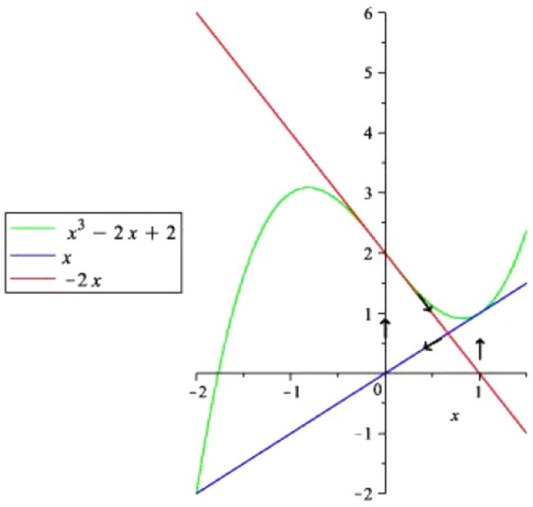

Fig. 4 Cyclic problem. Source: Adapted from [8]

(c) Cyclic problem

In some complex cases, there comes a time in the iterations when the process gets stuck between two points i.e. that one value leads to another and that new value leads back to the old one. This constant loop doesn't allow the process to move further hence there cannot be any roots found for the function (Fig. 4).

Algorithm of the Program

The flowchart on the following page describes how the program's processes run in order to calculate the roots of a function over an interval (Fig. 5).

The flow chart can be broken down into two flow charts for easier understanding and implementation (Fig. 6).

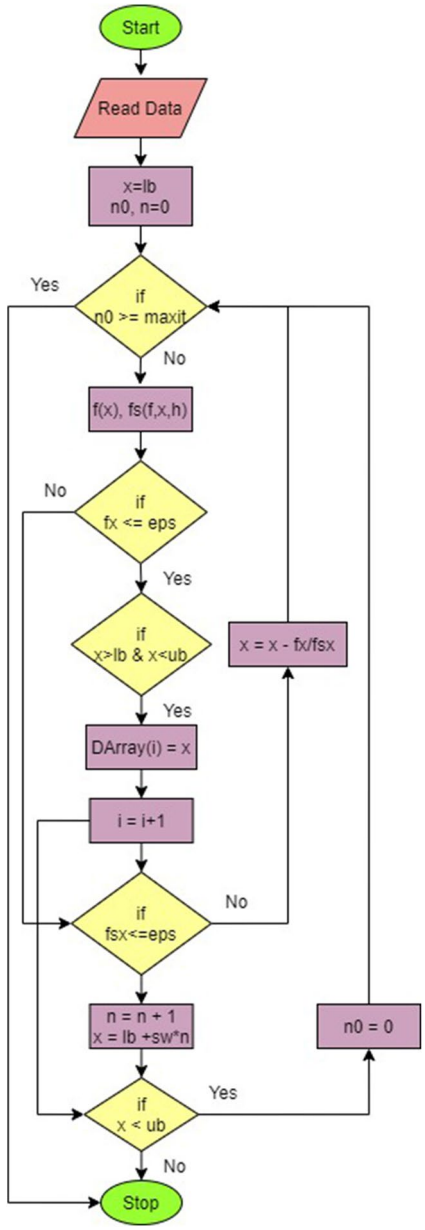

Fig. 5 Algorithm for Newton's method

Variables of the Program

In order to understand any program, it is necessary to first know what the variables inside actually stand for. The following table is an insight into the description of those variables (Table 1).

Error-Checking

Before writing the algorithm, it is necessary to first identify possible errors that could be problematic for the program. Methods to prevent them are then later incorporated into the program. The following errors were considered to be essential to be removed or to be dealt with:

- Lower bound should be different than the upper bound.
- Lower bound should be lesser than the upper bound.
- Step-width should be a non-negative number.
- The iterations should be lesser than the maximum allowable number of iterations.
- The maximum number of roots should be greater zero.

Comparison of Results

In order to test the credibility and accurateness of the program, it is necessary to compare it's results with the known thresholds. For the same reasons, the program was used for three given functions (3 cases) and the roots calculated by the program for those functions were compared those seen on a plotted graph. For plotting of the graph, online tools were used.

Case-1

The function for case-1 is given below:

$$\frac{\left(1-5x-2x^2\right)\times\tan\left(2x\right)}{\left(1-x\right)\sin\left(3x\right)\times\cos\left(4x\right)}.$$

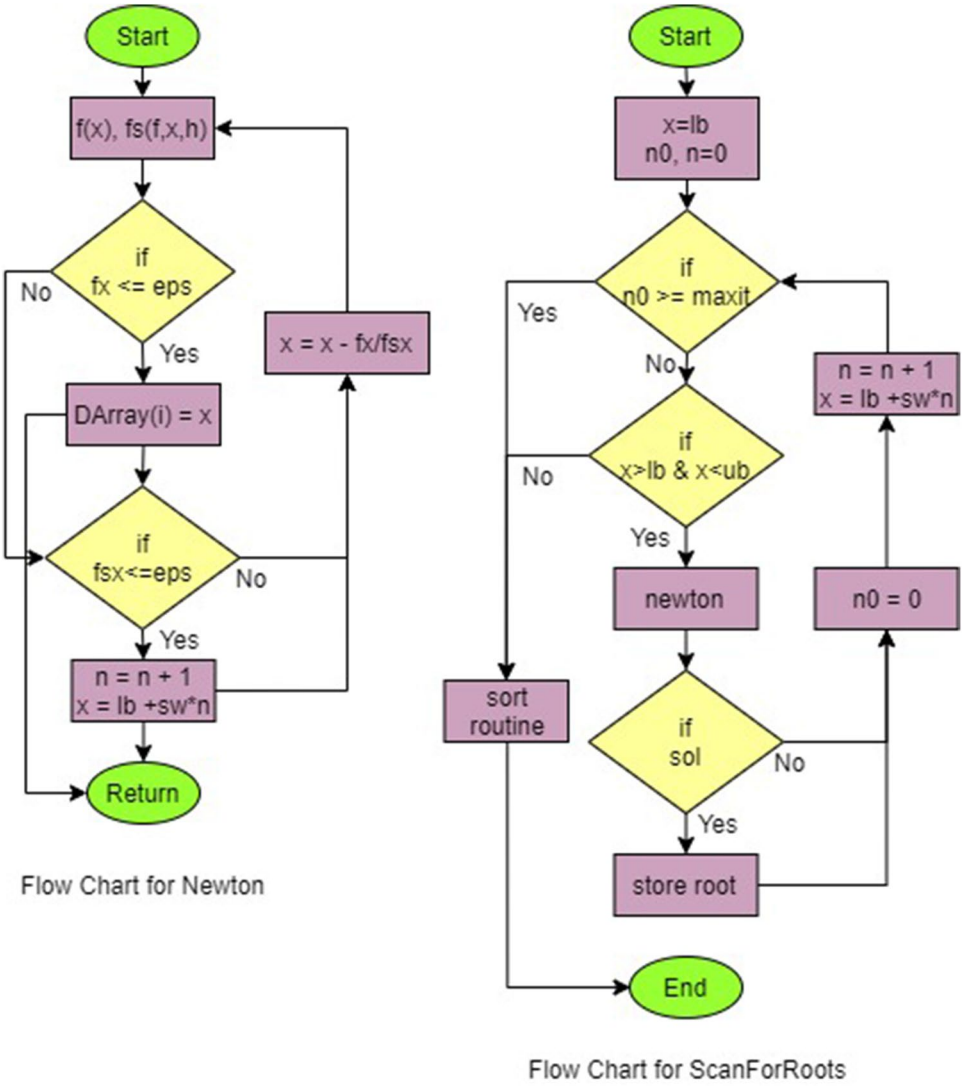

Fig. 6 Algorithm for Newton's method (broken down)

Table 1 Variables and their description

Variable	Description
rootsmax	Maximum no. of roots
lb	Lower bound
ub	Upper bound
sw	Step width
eps	Precision
h	*h* value for slope calculation
maxit	Maximum no. of iterations
x	Supposed root
fx	Value of function
fsx	Derivative of the function
le	Length of array

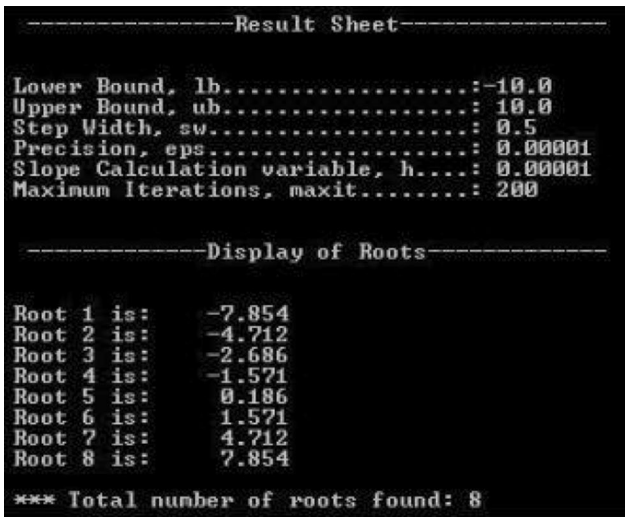

Fig. 7 Results by the program for Case-1

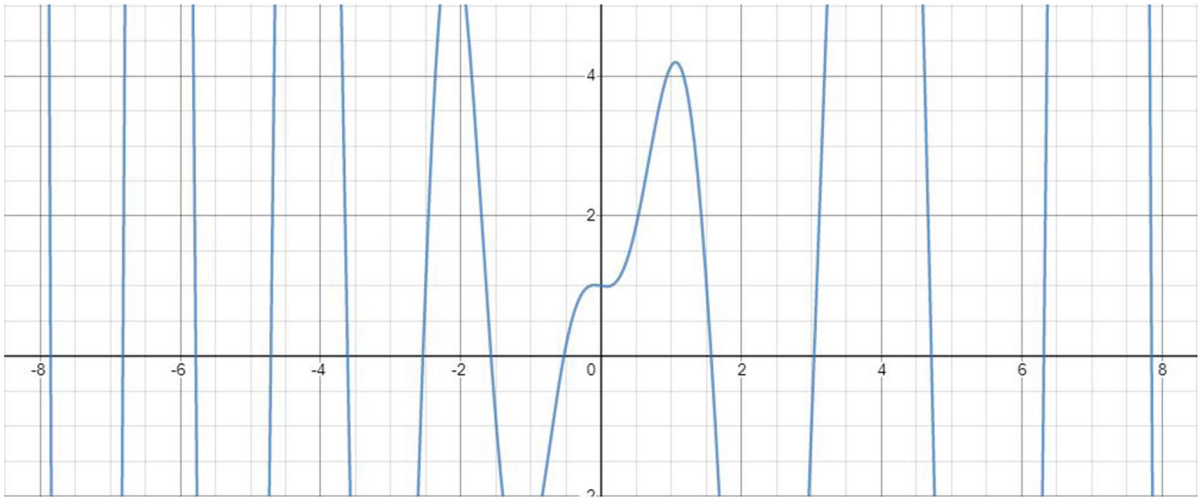

Fig. 8 Plot of the function in between the interval for Case-1

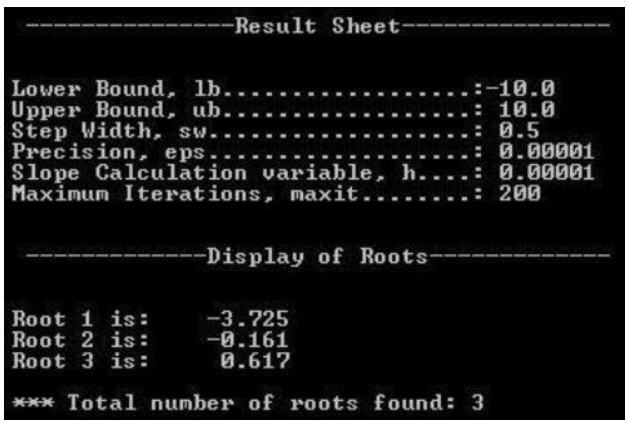

Fig. 9 Results by the program for Case-2

The results obtained by the program are shown by the figure below (Fig. 7).
The following graph shows the roots for the same interval as given in the program for case 1 (Fig. 8).

Case-2

The function for case-2 is given below:

$$\frac{x^5}{2} + 2x^4 + 3x^3 8x^2 - 5x - 1.$$

The results obtained by the program are shown by the figure below (Fig. 9).
The following graph shows the roots for the same interval as given in the program for case-2 (Fig. 10).

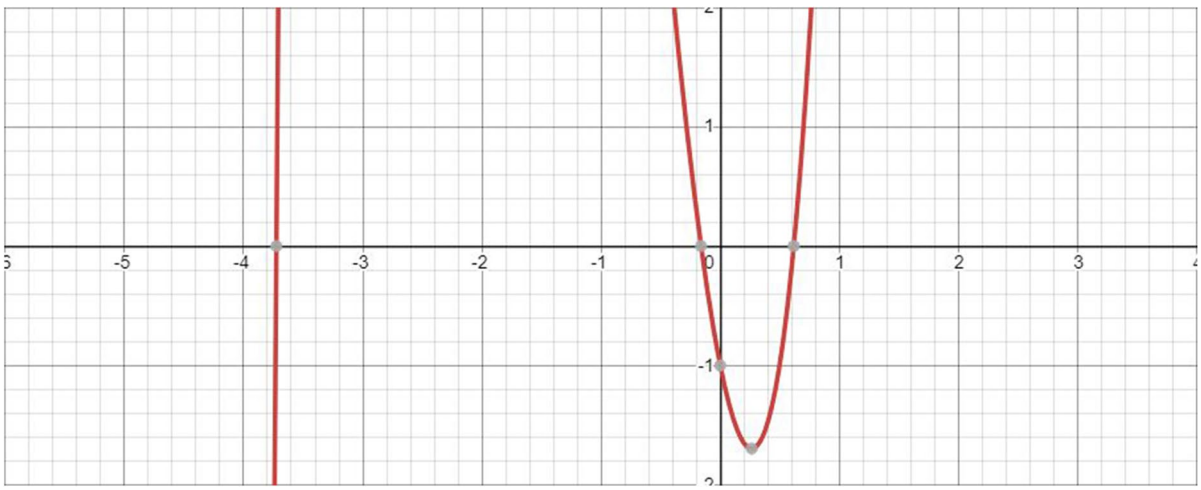

Fig. 10 Plot of the function in between the interval for Case-2

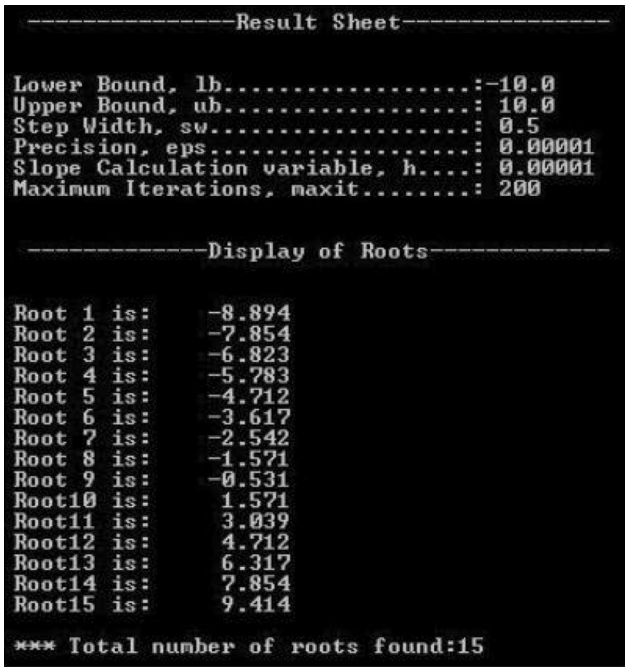

Fig. 11 Results by the program for Case-3

Case-3

The function for case-3 is given below:

$$e^{-x/5}((1-2x)\times\cos(3x))+e^{x/5}((1+2x)\times\sin(2x)).$$

The results obtained by the program are shown by the figure below (Fig. 11).

The following graph shows the roots for the same interval as given in the program for case-3 (Fig. 12).

Discussion and Conclusion

The previous section shows side by side, the results of the program and the actual roots of those functions. By comparison it can be easily said that the number of roots and their values match those shown on the graphs. Therefore, it is safe to say that the program is accurate and gives reliable results.

Finding the roots of complex equations can be very difficult and tedious at times. With this technique it can be

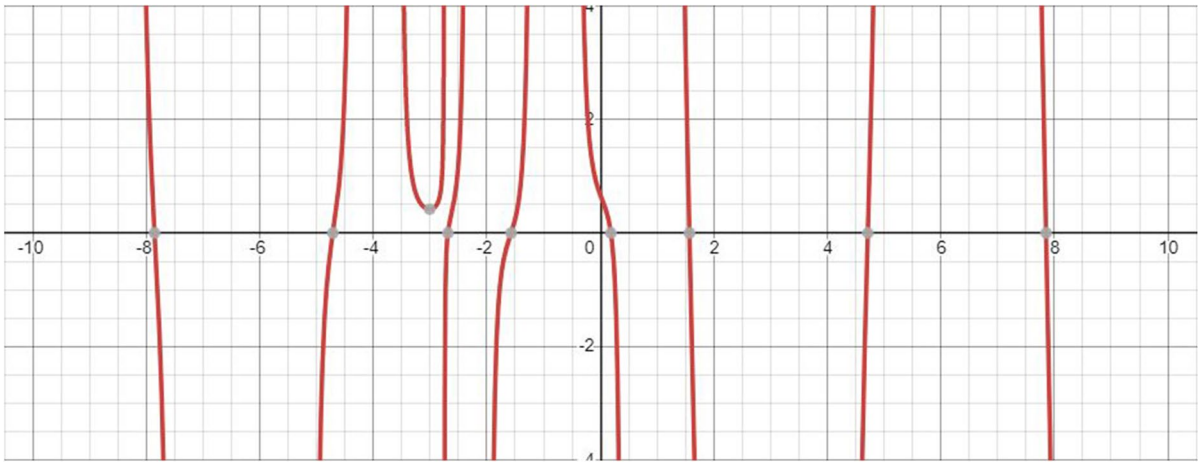

Fig. 12 Plot of the function in between the interval for Case-3

very easy to give good estimates of the results, provided that there are no errors possible during calculation within that interval.

Funding Open Access funding enabled and organized by Projekt DEAL.

Compliance with Ethical Standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Chill R. Three variations on Newton's method. Math Stud. 2008;79:215.
2. Galántai A. The theory of Newton's method. J Comput Appl Math. 2000;124(1–2):25–44. https://doi.org/10.1016/S0377-0427(00)00435-0.
3. Saheya B, Chen G, Sui Y, Wu C. A new Newton-like method for solving nonlinear equations. Springerplus. 2016. https://doi.org/10.1186/s40064-016-2909-7.
4. Dembo R, Eisenstat S, Steihaug T. Inexact Newton methods. SIAM J Numer Anal. 1982;19:400–8. https://doi.org/10.1137/0719025.
5. Koblitz N (1998) Calculus. University of Washington. 1998. https://www.ms.uky.edu/~carl/ma123/kob98/kob98htm/toc.html
6. Department of Biochemistry and Molecular Biophysics (2016) Roots of quadratic equations and the quadratic formula. 2016. https://www.biology.arizona.edu/biomath/tutorials/quadratic/roots.html
7. Collier N, Kaw A, Paul J, Keteltas M (2013) Zero slope—Newton–Raphson method. 2013. https://mathforcollege.com/nm/simulations/mws/03nle/mws_nle_sim_newpitzero.pdf
8. Burton A (2020) Newton's method and fractals. https://www.whitman.edu/documents/Academics/Mathematics/burton.pdf